CLAIMS

What we claim is:

1. A method of distributing a computer program module, the method including

distributing a computer program component which includes code defining functionality associated with the module and excludes version identification data for the module to execute the functionality under command from a master computer program; and

distributing an installation module which, when run on a computer, obtains the version identification data from the master computer program and combines the version identification data and the computer program component to define the computer program module.

2. The method of Claim 1, in which the master computer program is an operating system and the computer program module is a device driver, the master computer program being identifiable by the version identification data.

3. The method of Claim 2, in which the operating system is one of a Linux and a UNIX; operating system.

4. The method of Claim 3, in which the functionality included in the computer program component allows the computer program module to execute an application program interface (API) exported from the master computer program.

5. The method of Claim 3, which includes compiling the computer program component into an object file prior to distribution of the computer program module.

6. The method of Claim 5, which includes obtaining version identification data from the operating system and generating a version object file that includes the identification data.

7. The method of Claim 6, which includes linking the version object file and the computer program component.

8. The method of Claim 7, which includes obtaining a kernel specific address of a module list and passing the address to the computer program module.

9. The method of Claim 2, in which the device driver is one of a printer driver, a serial port device driver, an ethernet device driver, and a disk drive device driver.

10. The method of Claim 1, in which the installation module forms part of the computer program component.

11. A computer program product including a medium readable by a computer, the medium carrying instructions which, when executed by the computer, cause the computer to:

identify a computer program component which includes object code defining functionality associated with the product and excludes version identification data for the product to execute the functionality under command from a master computer program;

obtain the version identification data from the master computer program and combine the version identification data and the computer program component to define a computer program module; and

store the computer program module in memory.

12. The product of Claim 11, in which the master computer program is an operating system and the computer program module is a device driver, the master computer program being identifiable by the version identification data.

13. The product of Claim 12, in which the master computer program is one of a Linux and a UNIX, operating system.

14. The product of Claim 13, in which the functionality included in the computer program component allows the computer program module to execute at least one application program interface (API) exported from the master computer program.

15. The product of Claim 14, which includes obtaining version identification data from the operating system and generating a version object file that includes the version identification data.

16. The product of Claim 15, which includes linking the version object file and the computer program component to generate an object file that defines the computer program module.

17. The product of Claim 16, which obtains a kernel specific address of a module list and passes the address to the computer program product.

18. The product of Claim 17, in which the computer program product retrieves a module list export head and imports the required application program interfaces (APIs) ignoring the version identification data.

19. The product of Claim 13, in which the device driver is dynamically loaded

in a Linux kernel.

20. The product of Claim 11, in which the installation module forms part of the computer program component.

21. A computer program product including a medium readable by a computer, the medium carrying instructions which, when executed by the computer, cause the computer to:

define symbols to be imported from a Linux kernel, the symbols being uniquely associated with a particular version of the Linux kernel and used by the computer program product which operatively defines a device driver;

declare structures that describe application program interfaces (APIs) to be imported from the Linux kernel for operation of the device driver;

obtain the symbols that define identification data from the Linux kernel;

combine the symbols with driver code functionality provided by the computer program product ; and

dynamically import the device driver in the Linux kernel.

22. The product of Claim 21, which defines macros that build a linked list of

the symbols to be imported from the Linux kernel.

23. The product of Claim 21, which defines function stubs for registering the device driver.

24. The product of Claim 21, which defines a memory structure of a particular device for which the device driver is configured.

25. The product of Claim 24, which iteratively imports each symbol's kernel address and places the address into a local variable for use by the device driver.